

A practical approximation algorithm for solving massive instances of hybridization number

Leo van Iersel¹, Steven Kelk², Nela Lekić² and Celine Scornavacca³

¹ Centrum Wiskunde & Informatica (CWI), P.O. Box 94079,
1090 GB Amsterdam, The Netherlands, l.j.j.v.iersel@gmail.com

² Department of Knowledge Engineering (DKE), Maastricht University, P.O. Box
616, 6200 MD Maastricht, The Netherlands,

steven.kelk@maastrichtuniversity.nl, nela.lekic@maastrichtuniversity.nl

³ Institut des Sciences de l'Evolution (ISEM, UMR 5554 CNRS), Université
Montpellier II, Place E. Bataillon - CC 064 - 34095 Montpellier Cedex 5, France,
celine.scornavacca@univ-montp2.fr

Abstract. Reticulate events play an important role in determining evolutionary relationships. The problem of computing the minimum number of such events to explain discordance between two phylogenetic trees is a hard computational problem. In practice, exact solvers struggle to solve instances with reticulation number larger than 40. For such instances, one has to resort to heuristics and approximation algorithms. Here we present the algorithm CYCLEKILLER which is the first approximation algorithm that can produce solutions verifiably close to optimality for instances with hundreds or even thousands of reticulations. Theoretically, the algorithm is an exponential-time 2-approximation (or 4-approximation in its fastest mode). However, using simulations we demonstrate that in practice the algorithm runs quickly for large and difficult instances, producing solutions within one percent of optimality. An implementation of this algorithm, which extends the theoretical work of [14], has been made publicly available.

1 Introduction

1.1 Background

A phylogenetic tree is a model used in biology to represent the evolutionary history of a set \mathcal{X} of species (or *taxa*) [9,10]. They are trees whose leaves are bijectively labeled by \mathcal{X} and whose internal vertices represent the ancestors of the species set; they can be rooted or unrooted. Since in a rooted tree edges have a direction, the concepts of indegree and outdegree of a vertex are well defined. *Binary* rooted (phylogenetic) trees are rooted (phylogenetic) trees whose internal vertices have outdegree 2.

Some biological events such as hybridization, recombination and horizontal gene transfer cannot be modeled by a tree. To represent an event

in which a species derives its genes from different ancestors, trees are generalized to allow vertices with indegree two or higher, known as *reticulations*. This model is called a *rooted phylogenetic network*. For detailed background information we refer the reader to [11,13,15]. Note that there also exist models based on *unrooted* phylogenetic networks but these are not the subject of this article.

Although phylogenetic networks are more general than phylogenetic trees, trees are still often the basic building blocks from which they are constructed. Specifically, there are many techniques available for constructing gene trees. However, when more genes are analyzed, topological conflicts between individual gene phylogenies can arise for methodological or biological reasons (i.e. aforementioned reticulate phenomena such as hybridization). This led computational biologists to try and quantify the amount of reticulation that is needed to simultaneously explain two trees.

To state this problem more formally, we say that a phylogenetic network N on \mathcal{X} *displays* a phylogenetic tree T on \mathcal{X} if T can be obtained from a subtree of N by contracting edges. (Informally this means that T can be obtained from N by, for each reticulation vertex of N , “switching off” all but one of its incoming edges and then suppressing all indegree-1 outdegree-1 vertices). Given two rooted binary phylogenetic trees T_1 and T_2 on \mathcal{X} , the problem then becomes to determine the minimum number of reticulations contained in a phylogenetic network N on \mathcal{X} displaying both trees. The value we are minimizing is often called the *hybridization number* and instead of the term phylogenetic network, the term *hybridization network* is often used.

It is known that the problem of computing hybridization numbers is both NP-hard and APX-hard [4], but it is not known whether it is in APX (i.e. whether it admits a polynomial-time approximation algorithm that achieves a constant approximation ratio).

Until recently, most research on the hybridization number of two binary trees had focused on the question of how to exactly compute this value using fixed parameter tractable (FPT) algorithms. For an introduction to FPT we refer to [8]. Algorithmic progress has been considerable in this area, with various authors reporting increasingly sophisticated FPT algorithms where the parameter in question is the hybridization number r of the two trees [3,5,7,18]. The fastest algorithms currently implemented are the algorithm available inside the package DENDROSCOPE [12], based on [1], and the algorithm HYBRIDNET [5]. The fastest theoretical FPT

algorithm has running time $O(3.18^r n)$ [18], where n is the number of taxa in the trees.

Such FPT algorithms do, however, have their limits. For NP-hard problems such as hybridization number, and under the assumption that $P \neq NP$, the running time still grows exponentially in r , albeit usually at a slower rate than algorithms that have a running time of the form $n^{f(r)}$ where f is some function of r . In practice this means that existing algorithms struggle to terminate for instances with hybridization number larger than 40.

For such large and/or difficult trees one can try to generate heuristic or approximate solutions, but how far are such solutions from optimality? In [14] we showed that the news is worrying. Indeed, we showed that polynomial-time constant-ratio approximation algorithms exist if and only if such algorithms exist for the problem Directed Feedback Vertex Set (DFVS). However, DFVS is one of the most well-studied problems in combinatorial optimization and to this day it is unknown if it permits such an algorithm. Pending a major breakthrough in computer science, it therefore seems difficult to build efficient algorithms which approximate hybridization number well. On the positive side, we showed that in polynomial time an algorithm with approximation ratio $O(\log r \log \log r)$ is possible. However, this algorithm is purely of theoretical interest and is not useful in practice.

1.2 A new algorithm: CycleKiller

In this article we extend the theoretical work of [14] slightly and give it a practical twist to yield a fast approximation algorithm (which we have made publicly available as the program CYCLEKILLER) that achieves a low constant ratio approximation, even for massive trees where the hybridization number runs into hundreds or even thousands.

The worst-case running time of this approximation algorithm is exponential. However, as we demonstrate with experiments, the running time of the algorithm is in practice extremely fast and, when the hybridization number is large, typically orders of magnitude faster than HYBRID-NET or the algorithm in DENDROSCOPE. Of course, those algorithms attempt to compute optimum solutions, whereas our algorithm only gives approximate solutions. Nevertheless, our experiments show that when CYCLEKILLER is run in its most accurate mode of operation an approximation ratio very close to 1 is not unusual, suggesting that the algorithm often produces solutions close to optimality and well within the worst-case approximation guarantee.

Specifically, we describe an algorithm with approximation ratio $c(d + 1)$ for the hybridization number problem by combining a c -approximation for the problem MAF (Maximum Agreement Forest) (see e.g. [16,18]) with a d -approximation for the problem DFVS. Both these problems are NP-hard so polynomial-time algorithms attaining $c = 1$ or $d = 1$ are not realistic. Nevertheless, there exist extremely fast FPT algorithms for solving MAF exactly (i.e. $c = 1$), the fastest is rSPR by Whidden, Beiko and Zeh [17,19]. Moreover, we observe that the type of DFVS instances that arise in practice can easily be solved using Integer Linear Programming (ILP) (and freely-available ILP solver technology such as GLPK), so $d = 1$ is also often possible. Combining these two exact approaches gives us an exponential-time approximation algorithm with worst-case approximation ratio 2 that for large instances still runs extremely quickly; this is the **2-approx** option of CYCLEKILLER. In practice, we have observed that the upper bound of 2 is often pessimistic, with much better approximation ratios observed in experiments (1.003 for the simulations presented in this article). We find that this algorithm already allows us to cope with much bigger trees than HYBRIDNET or the algorithm in DENDROSCOPE.

Nevertheless, for truly massive trees it is often not feasible to have $c = 1$. Fortunately there exist linear-time algorithms which achieve $c = 3$ [18]. This, coupled with the fact that (even for such trees) it remains feasible to use an exact ($d = 1$) solver for DFVS, means that in practice we achieve a 4-approximation for gigantic trees; this is the **4-approx** option of CYCLEKILLER. Again, the ratio of 4 is a worst-case bound and we suspect that in practice we are doing much better than 4. However, this cannot be experimentally verified due to the lack of good lower bounds for such massive instances. In any case, the main advantage of the 4-approximation is that it can without too much effort cope with trees with hundreds or thousands of taxa and hybridization number of a similar order of magnitude. An implementation of CYCLEKILLER and accompanying documentation can be downloaded from <http://skelk.sdf-eu.org/cyclekiller>. Networks created by the algorithm can be viewed in DENDROSCOPE.

1.3 Theoretical and practical significance

We have described, implemented and made publicly available an algorithm with two desirable qualities: it terminates quickly even for massive instances of hybridization number and it gives a non-trivial guarantee of proximity to optimality. This is the first algorithm with such properties. The algorithm is a non-trivial marriage of MAF and DFVS solvers (both

exact and approximate), meaning that further advances in solving MAF and DFVS will directly lead to improvements in CYCLEKILLER.

This article also improves the theoretical work given in [14], which also proposed using DFVS but beginning from a trivial Agreement Forest (AF) known as a *chain forest*. Here we use a more intelligent starting point: an (approximate) MAF, and it is this insight which makes a 2-approximation (rather than the 6-approximation implied by [14]) possible when using an exact DFVS solver. Other articles have also had the idea of cycle-breaking in AFs: the advanced FPT algorithm of Whidden et al [18] – that, as far as we know, has not been implemented – and the algorithm HYBRIDNET by Chen et al. [5] (also see [6]). However, both algorithms start the cycle-breaking from many starting points. In contrast, our algorithm requires only a *single* starting point, i.e. an (approximate) solution to MAF.

2 Preliminaries

Let \mathcal{X} be a finite set (e.g. of species). A *rooted phylogenetic \mathcal{X} -tree* is a rooted tree with no vertices with indegree 1 and outdegree 1, a root with indegree 0 and outdegree at least 2, and leaves bijectively labelled by the elements of \mathcal{X} . We identify each leaf with its label and use $L(T)$ to refer to the leaf set (or label set) of T . A rooted phylogenetic \mathcal{X} -tree is called *binary* if each nonleaf vertex has outdegree two. We henceforth call a rooted, binary phylogenetic \mathcal{X} -tree a *tree* for short. For a tree T and a set $\mathcal{X}' \subset \mathcal{X}$, we use the notation $T(\mathcal{X}')$ to denote the minimal subtree of T that contains all elements of \mathcal{X}' and $T|\mathcal{X}'$ denotes the result of suppressing all indegree-1 outdegree-1 vertices in $T(\mathcal{X}')$.

We define a *forest* as a set of trees. Each element of a forest is called a *component*. Let T be a tree and \mathcal{F} a forest. We say that \mathcal{F} is a *forest for T* if $T|L(F)$ is isomorphic to F for all $F \in \mathcal{F}$ and the trees $\{T(L(F)), F \in \mathcal{F}\}$ are vertex-disjoint subtrees of T whose leaf-set union equals $L(T)$. If T_1 and T_2 are two trees, then a forest \mathcal{F} is an *agreement forest* of T_1 and T_2 if it is a forest for T_1 and T_2 . The number of components of \mathcal{F} is denoted $|\mathcal{F}|$.

We define *cleaning up* a directed graph as repeatedly suppressing indegree-1 outdegree-1 vertices, removing indegree-0 outdegree-1 vertices and removing unlabelled outdegree-0 vertices until no such operation is possible. Observe that, if \mathcal{F} is a forest for T , \mathcal{F} can be obtained from T by removing $|\mathcal{F}| - 1$ edges and cleaning up.

Problem: Maximum Agreement Forest (MAF)

Instance: Two rooted, binary phylogenetic trees T_1 and T_2 .

Solution: An agreement forest \mathcal{F} of T_1 and T_2 .

Objective: Minimize $|\mathcal{F}| - 1$.

The directed graph $IG(T_1, T_2, \mathcal{F})$, called the *inheritance graph*, is the directed graph whose vertices are the components of \mathcal{F} and which has an edge (F, F') precisely if either

- there is a directed path in T_1 from the root of $T_1(L(F))$ to the root of $T_1(L(F'))$ or;
- there is a directed path in T_2 from the root of $T_2(L(F))$ to the root of $T_2(L(F'))$.

An agreement forest \mathcal{F} of T_1 and T_2 is called an *acyclic agreement forest* if the graph $IG(T_1, T_2, \mathcal{F})$ is acyclic. A *maximum acyclic agreement forest* (MAAF) of T_1 and T_2 is an acyclic agreement forest of T_1 and T_2 with a minimum number of components.

Problem: Maximum Acyclic Agreement Forest (MAAF)

Instance: Two rooted, binary phylogenetic trees T_1 and T_2 .

Solution: An acyclic agreement forest \mathcal{F} of T_1 and T_2 .

Objective: Minimize $|\mathcal{F}| - 1$.

We use $MAF(T_1, T_2)$ and $MAAF(T_1, T_2)$ to denote the optimal solution value of the problem MAF and MAAF respectively, for an instance T_1, T_2 .

A *rooted phylogenetic network* on \mathcal{X} is a directed acyclic graph with no vertices with indegree 1 and outdegree 1 and leaves bijectively labelled by the elements of \mathcal{X} . Rooted phylogenetic networks, which are sometimes also called hybridization networks, will henceforth be called *networks* for short in this paper. A tree T on \mathcal{X} is *displayed* by a network N if T can be obtained from a subtree of N by contracting edges. A *reticulation* is a vertex v with $\delta^-(v) \geq 2$ (with $\delta^-(v)$ denoting the indegree of v). The *reticulation number* (sometimes also called hybridization number) of a network N with root ρ is given by

$$r(N) = \sum_{v \neq \rho} (\delta^-(v) - 1).$$

It was shown that the optimum to MAAF is equal to the optimum of the following problem [2].

Problem: MINIMUMHYBRIDIZATION

Instance: Two rooted, binary phylogenetic trees T_1 and T_2 .

Solution: A rooted phylogenetic network N that displays T_1 and T_2 .

Objective: Minimize $r(N)$.

Moreover, it was shown that, for two trees T_1, T_2 , *any* acyclic agreement forest for T_1 and T_2 with $k + 1$ components can be turned into a phylogenetic network that displays T_1 and T_2 and has reticulation number k , and vice versa. Thus, any approximation for MAAF gives an approximation for MINIMUMHYBRIDIZATION.

A *feedback vertex set* of a directed graph is a subset of the vertices that contains at least one vertex of each directed cycle. Equivalently, a subset of the vertices of a directed graph is a *feedback vertex set* if removing these vertices from the graph makes it acyclic.

Problem: Directed Feedback Vertex Set (DFVS)

Instance: A directed graph D .

Goal: Find a feedback vertex set of D of minimum size.

3 Main result

We show how MAAF can be approximated by combining algorithms for MAF and DFVS. In particular, we will prove the following theorem.

Theorem 1. *If there exists a c -approximation for MAF and a d -approximation for DFVS, then there exists a $(c+1)d$ -approximation for MAAF (and thus for MINIMUMHYBRIDIZATION).*

Suppose there exists a c -approximation for MAF. Let T_1 and T_2 be two trees and let M be an agreement forest returned by the algorithm. Then,

$$|M| - 1 \leq c \cdot \text{MAF}(T_1, T_2) \leq c \cdot \text{MAAF}(T_1, T_2). \quad (1)$$

An M -*splitting* is an acyclic agreement forest that can be obtained from M by removing edges and cleaning up.

Lemma 1. *Let T_1 and T_2 be two trees and M an agreement forest of T_1 and T_2 . Then, there exists an M -splitting of size at most $\text{MAAF}(T_1, T_2) + |M|$.*

Proof. Consider a maximum acyclic agreement forest F of T_1 and T_2 . For $i \in \{1, 2\}$, F can be obtained from T_i by removing a set of edges, say E_F^i , and cleaning up. Moreover, also M can be obtained from T_i by removing a set of edges, say E_M^i , and cleaning up.

Now consider the forest S obtained from T_1 by removing $E_M^1 \cup E_F^1$ and cleaning up. Then,

- S is an agreement forest of T_1 and T_2 because it can be obtained from T_2 by removing edges $E_M^2 \cup E_F^2$ and cleaning up;
- S is acyclic because it can be obtained by removing edges from F , which is acyclic, and cleaning up;
- S can be obtained from M by removing edges and cleaning up.

Hence, S is an M -splitting. Furthermore, $|S| \leq |E_F^1| + |E_M^1| + 1$. The lemma follows since $|E_F^1| = \text{MAAF}(T_1, T_2)$ and $|M| = E_M^1 + 1$. \square

Let $\text{OptSplitting}_{T_1, T_2}(M)$ denote the size of a minimum-size M -splitting. Combining Lemma 1 and equation (1), we obtain

$$\text{OptSplitting}_{T_1, T_2}(M) - 1 \leq (c + 1)\text{MAAF}(T_1, T_2) \quad (2)$$

We will now show how to find an approximation for the problem of finding an optimal M -splitting. We do so by reducing the problem to DFVS. We construct an input graph D for DFVS as follows. For every vertex of M that has outdegree 2 (in M), we create a vertex in D . There is an edge in D from a vertex u to a vertex v precisely if in either T_1 or T_2 (or in both) there is a directed path from u to v . We claim the following.

Lemma 2. *A subset V' of the vertices of D is a feedback vertex set of D if and only if removing V' from M makes it an acyclic agreement forest.*

Proof. We show that $D \setminus V'$ has a directed cycle if and only if the inheritance graph of $M \setminus V'$ has a directed cycle.

To prove that, first suppose that there is a cycle $v_1, v_2, \dots, v_k = v_1$ in the inheritance graph of $M \setminus V'$. The vertices in the inheritance graph of $M \setminus V'$ correspond to the roots of the components of $M \setminus V'$. Since these roots have outdegree 2 in $M \setminus V'$, they had outdegree 2 in M , and are thus vertices of D . So the vertices v_1, v_2, \dots, v_k that form the cycle are vertices of D . Since these vertices are in the inheritance graph of $M \setminus V'$, they can not be in V' and so they are vertices of $D \setminus V'$. The reachability relation between these vertices in $D \setminus V'$ is the same as in the inheritance graph of $M \setminus V'$. So, the vertices v_1, v_2, \dots, v_k form a cycle in $D \setminus V'$.

Now suppose that there is a cycle $w_1, w_2, \dots, w_k = w_1$ in $D \setminus V'$. Each of the vertices w_1, w_2, \dots, w_k is a vertex with outdegree-2 in M . Some of them might be roots of components, while others are not. However, observe that if there is a directed path from a vertex u to a vertex v in T_1 (or in T_2) then there is also a directed path from the root of the component of $M \setminus V'$ that contains u to the root of the component of $M \setminus V'$ that contains v . Hence, there is a directed cycle in the inheritance graph of $M \setminus V'$, formed by the roots of the components of $M \setminus V'$ that contain w_1, w_2, \dots, w_k . \square

Suppose that there exists a d -approximation for DFVS. Let FVS be a feedback vertex set returned by this algorithm and let MFVS be a minimum feedback vertex set. Then, removing the vertices of MFVS from M gives an optimal M -splitting. Furthermore, $\text{OptSplitting}_{T_1, T_2}(M) = |M| + |\text{MFVS}|$. This is because for every vertex in a cycle C , its parent in M must participate in some cycle that contains elements of C . So if we start by removing the root of the component we are splitting and subsequently remove only those vertices whose parents have already been removed we see that we add at most one component per vertex. In fact, because vertices of D all have out-degree 2 in M , we add exactly one component per vertex.

By removing the vertices of FVS from M , we obtain an acyclic agreement forest \mathcal{F} such that

$$\begin{aligned} |\mathcal{F}| - 1 &= |M| + |FVS| - 1 \\ &\leq |M| + d \cdot |\text{MFVS}| - 1 \\ &\leq d(|M| + |\text{MFVS}| - 1) \\ &= d(\text{OptSplitting}_{T_1, T_2}(M) - 1) \\ &\leq d(c + 1)\text{MAAF}(T_1, T_2), \end{aligned}$$

where the last inequality follows from equation (2). Thus, \mathcal{F} is a $(c + 1)d$ -approximation to MAAF, which concludes the proof of Theorem 1.

4 Practical experiments

To assess the performance of CYCLEKILLER, a simulation study was undertaken. We generated 3 synthetic datasets, an *easy*, a *medium* and a *hard* one, containing respectively 800, 640 and 640 pairs of rooted binary phylogenetic trees.

The easy data set was created by varying two parameters, namely the number of taxa n and the number of rSPR-moves k used to obtain the second tree from the first (note that this number is an upper bound on the actual rSPR distance). The 800 pairs of rooted binary phylogenetic trees were created by varying n in $\{20, 50, 100, 200\}$ and k in $\{5, 10, \dots, 25\}$, and then creating 40 different instances per each combination of parameters. Each pair (T_1, T_2) of rooted binary phylogenetic trees for a given set of parameters n and k is created as follows: The first tree T_1 on $\mathcal{X} = \{x_1, \dots, x_n\}$ is generated by first creating a set of n leaf vertices bijectively labeled by the set \mathcal{X} . Then, two vertices u and v , both with indegree 0, are randomly picked and a new vertex w , along with two new edges (w, u)

and (w, v) , is created. This is done until only one vertex with no ancestor, the root, is present. The second tree T_2 is obtained from T_1 by applying k rSPR-moves. The medium and the hard data sets were generated in the same way as the easy one, but for different choices of the parameters: n in $\{50, 100, 200, 300\}$ and k in $\{15, 25, 40, 55\}$ for the medium one and n in $\{100, 200, 400, 500\}$ and k in $\{40, 60, 80, 100\}$ for the hard one.

The exact hybridization number has been computed by HYBRIDNET [5], available from <http://www.cs.cityu.edu.hk/~lwang/software/Hn/treeComp.html> or with DENDROSCOPE [12], available from <http://www.dendroscope.org>. We will refer to these algorithms as the *exact algorithms*. Each instance has been run on a single core of an Intel Xeon E5506 processor.

Each run that took more than one hour was aborted. For each instance, we ran our program with the option **2-approx**, and, in case the latter did not finish within one hour, we ran it again, this time using the option **4-approx**, always with a one-hour limit. We used the program RSPR v1.03 [17,19] to solve or approximate MAF and GLPK v4.47 (<http://www.gnu.org/software/glpk/>) to solve a simple ILP formulation of DFVS.

For all instances of the easy data set, CYCLEKILLER finished with the **2-approx** option within the one hour limit, while for 33 instances the exact algorithms were unable to compute the hybridization number. Note that, even for “easy” instances, computing the exact hybridization number can take a very long time. To give the reader an idea, for 9 runs of the easy data, DENDROSCOPE and HYBRIDNET did not complete within 10 days. Table 1 shows a summary of the results. It can be seen that CYCLEKILLER was much faster than the exact algorithms. Moreover, for 96.6% of the instances for which an exact algorithm could find a solution, CYCLEKILLER also found an optimal solution. While the theoretical worst-case approximation ratio of the **2-approx** option of CYCLEKILLER is 2, in our experiments it performed very close to a 1-approximation.

For the medium data set, CYCLEKILLER finished with the **2-approx** option for 613 instances, and for the remaining ones with the **4-approx** option. The exact algorithms could compute the hybridization number for only 199 instances (out of 640). For 97.5% of these instances, CYCLEKILLER also found an optimal solution, but with a much better running time. Regarding the hard data set, 444 runs were completed with the **2-approx** option and for the remaining ones we were able to use the **4-approx** option within the given time constraint. Unfortunately, the exact algorithms were unable to compute the hybridization number for any tree-pair of this data set and hence we could not compute the average

Dataset	Total runs	Exact algorithms		CYCLEKILLER					
		Completed	Running time (RT)	2-approx		4-approx		Average appr. ratio	Opt. found
				Compl.	RT	Compl.	RT		
Easy	800	767	13m18s	800	3s	-	-	1.003	96.6%
Medium	640	199	42m52s	613	3m32s	27	<1 s	1.002	97.5%
Hard	640	0	1h	440	21m11s	200	1.5 s	-	-

Table 1. Experimental results. The third column indicates for how many instances at least one exact algorithm finished within one hour. The fifth column indicates for how many instances the **2-approx** option of CYCLEKILLER finished within one hour. For the remaining instances, the **4-approx** option finished within one hour, as can be seen from the seventh column. The average running time for the **2-approx** and the **4-approx** are reported respectively in the sixth and eighth column. The average approximation ratio (ninth column) is taken over all instances for which at least one exact method finished. The last column indicates the percentage of those instances for which CYCLEKILLER found an optimal solution.

approximation ratios. Over all our experiments, the maximum hybridization number that the exact algorithms could handle was 25.⁴ In contrast, the **2-approx** option of CYCLEKILLER could be used for instances for which the size of a MAF was up to 97, and thus for instances for which the hybridization number was at least 97.

To find the limits of the **4-approx** option of CYCLEKILLER, we also tested it on randomly generated trees. On a normal laptop, it could construct networks with up to 10,000 leaves and up to 10,000 reticulations within 10 minutes. Since the number of reticulations found is at most four times the optimal hybridization number, this implies that the **4-approx** option of CYCLEKILLER can handle hybridization numbers up to at least 2,500. These randomly generated trees are, however, biologically meaningless and, therefore, we conducted the extensive experiment described above on trees generated by rSPR moves.

5 Acknowledgements

We thank Simone Linz and Leen Stougie for fruitful discussions. Leo van Iersel and Nela Lekić were supported by Veni and Vrije Competitie grants of The Netherlands Organisation for Scientific Research (NWO). This publication is the contribution no. 2012-XXX of the Institut des Sciences de l’Evolution de Montpellier (ISE-M, UMR 5554). This work has been partially funded by the ANCESTROME project ANR-10-IABI-0-01 and it has benefited from the ISE-M computing facilities.

⁴ In [1], it has been shown that this number can go up to 40 when running Dendroscope on a similar processor but allocating all cores for one instance, i.e. exploiting the possibilities of parallel computation of this implementation.

References

1. B. Albrecht, C. Scornavacca, A. Cenci, and D.H. Huson. Fast computation of minimum hybridization networks. *Bioinformatics*, 28(2):191–197, 2012.
2. M. Baroni, S. Grünwald, V. Moulton, and C. Semple. Bounding the number of hybridisation events for a consistent evolutionary history. *Mathematical Biology*, 51:171–182, 2005.
3. M. Bordewich, S. Linz, K. St. John, and C. Semple. A reduction algorithm for computing the hybridization number of two trees. *Evolutionary Bioinformatics*, 3:86–98, 2007.
4. M. Bordewich and C. Semple. Computing the minimum number of hybridization events for a consistent evolutionary history. *Discrete Applied Mathematics*, 155(8):914–928, 2007.
5. Z-Z. Chen and L. Wang. Hybridnet: a tool for constructing hybridization networks. *Bioinformatics*, 26(22):2912–2913, 2010.
6. Z-Z. Chen and L. Wang. Algorithms for reticulate networks of multiple phylogenetic trees. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 9(2):372–384, 2012.
7. J. Collins, S. Linz, and C. Semple. Quantifying hybridization in realistic time. *Journal of Computational Biology*, 18:1305–1318, 2011.
8. J. Flum and M. Grohe. *Parameterized Complexity Theory*. Springer, 2006.
9. O. Gascuel, editor. *Mathematics of Evolution and Phylogeny*. Oxford University Press, Inc., 2005.
10. O. Gascuel and M. Steel, editors. *Reconstructing Evolution: New Mathematical and Computational Advances*. Oxford University Press, USA, 2007.
11. D.H. Huson, R. Rupp, and C. Scornavacca. *Phylogenetic Networks: Concepts, Algorithms and Applications*. Cambridge University Press, 2011.
12. D.H. Huson and C. Scornavacca. Dendroscope 3 - a program for computing and drawing rooted phylogenetic trees and networks. In preparation. Software available from: www.dendroscope.org, 2011.
13. D.H. Huson and C. Scornavacca. A survey of combinatorial methods for phylogenetic networks. *Genome Biology and Evolution*, 3:23–35, 2011.
14. S.M. Kelk, L.J.J. van Iersel, N. Lekić, S. Linz, C. Scornavacca, and L. Stougie. Cycle killer... qu’est ce que c’est? on the comparative approximability of hybridization number and directed feedback vertex set. Submitted, preliminary version arXiv:1112.5359v1 [math.CO].
15. L. Nakhleh. *The Problem Solving Handbook for Computational Biology and Bioinformatics*, chapter Evolutionary phylogenetic networks: models and issues. Springer, 2009.
16. E.M. Rodrigues, M.F. Sagot, and Y. Wakabayashi. The maximum agreement forest problem: Approximation algorithms and computational experiments. *Theoretical Computer Science*, 374((1-3):91–110, 2007.
17. C. Whidden. <http://kiwi.cs.dal.ca/Software/RSPR>.
18. C. Whidden, R. G. Beiko, and N. Zeh. Fixed-parameter and approximation algorithms for maximum agreement forests. Submitted, preliminary version arXiv:1108.2664v1 [q-bio.PE].
19. C. Whidden, R.G. Beiko, and N. Zeh. Fast FPT algorithms for computing rooted agreement forests: Theory and experiments. In *Proceedings of the 9th International Symposium on Experimental Algorithms, SEA 2010*, volume 6049 of *Lecture Notes in Computer Science*, pages 141–153. 2010.